CLAIMS

What is claimed is:

1. A disk scheduling system, comprising:

at least one queue to hold a subset of Input/Output requests directed to a storage medium; and

a scheduling component that employs a predetermined number of the requests within a round to provide a particular latency level and maintain a particular throughput level in connection with storage medium updates.

2. The system of claim 1, the scheduling component is associated with a categorizer that automatically directs the requests one or more classes of queues.

3. The system of claim 2, the scheduling component includes a queue of predetermined size that receives the requests from the categorizer.

4. The system of claim 1, the storage medium includes at least one of a hard disk, floppy disk, memory stick, compact disk, and a memory that is scheduled to be accessed from a computer.

5. The system of claim 1, the scheduling component dynamically calculates a determined amount of bounded latency in order to perform I/O scheduling for selected tasks.

6. The system of claim 1, further comprising at least one of a periodic queue and an aperiodic queue.

7. The system of claim 6, the periodic queue is arranged in an Earliest-Deadline-first ordering.

8. The system of claim 6, the aperiodic queue is arranged according to criticality classes.

9. The system of claim 8, the criticality classes include at least one of critical, high, interactive, normal, background, low, and idle.

10. The system of claim 8, the aperiodic queue is arranged according to a First-In-First-Out (FIFO) ordering.

11. The system of claim 6, further comprising a sweep queue to process the periodic queue and the aperiodic queue and to access the storage medium.

12. The system of claim 11, the sweep queue is managed according to a C-LOOK component and arranged in C-LOOK order.

13. The system of claim 6, the periodic queue is associated with periodic I/O associated with multimedia applications.

14. The system of claim 6, further comprising a component to process periodic I/O streams having performance parameters that are checked in view of system resources.

15. The system of claim 6, further comprising a component that analyzes a periodic I/O stream in view of system capabilities and aborts a request if the capabilities are exceeded.

16. The system of claim 15, the component is an admission control component. (not shown) enforces that respective parameters of the periodic I/O are not exceeded.

17. The system of claim 16, the admission controller includes a deadline component to assign a deadline or timeframe to a request in which the request is to be completed.

18. The system of claim 1, further comprising the following equation to determine access time to the storage medium:

$$service(n) = n\left(time\_seek\left(\frac{Cylinders}{N} \right) + time_{transfer} + time_{rotation} + time_{controller} \right) + time_{sweep}$$

wherein n is an integer.

19. A computer readable medium having computer readable instructions stored thereon for implementing the scheduling component and the queue of claim 1.

20. A method to schedule requests to a storage medium, comprising:
determining a predetermined number of requests for a set of requests;
determining a latency parameter associated with the set of requests; and

updating a storage medium based upon the latency parameter and a desired throughput for the set of requests.

21. The method of claim 20, further comprising automatically separating Input/Output streams associated with the set of requests into at least one of periodic streams and aperiodic streams.

22. The method of claim 21, further comprising:
arranging the periodic streams in an Earliest Deadline First ordering; and
arranging the aperiodic streams in a First-In-First-Out ordering.

23. The method of claim 22, further comprising applying a C-LOOK algorithm to the periodic streams and the aperiodic streams to access a storage medium.

24. The of claim 20, further comprising applying functionality associated with the following instructions to access the storage medium:

```
while periodic-queue ≠Ø and periodic-slots ≠0 and
    DEADLINE (periodic-queue) < ROUND_TIME do
  INSERT_LBA_ORDER (sweep-queue,
    REMOVE_EDF_ORDER (periodic-queue))
  periodic-slots = periodic-slots - 1
end

for priority = High downto Idle do
   while priority-list [priority].slots ≠0
    INSERT_LBA_ORDER (sweep-queue,
      REMOVE_FIFO_ORDER (priority-queue [priority])
    priority-list [priority].slots = priority-list [priority].slots - 1
   end
end

while sweep-queue ≠Ø do
    SCHEDULE (REMOVE_MINMUM_LBA (sweep-queue))
end
```

25. A method to perform disk updates, comprising:

automatically determining latency requirements for a round of requests;

automatically adjusting a size for the round of requests; and

updating a storage media with the round of requests.

26. The method of claim 27, further comprising:

determining bandwidth requirements for the round of requests; and

automatically prioritizing the round of requests based upon the determined bandwidth requirements.

27. The method of claim 28, further comprising dynamically monitoring the bandwidth and the latency requirements.

28. A system to facilitate disk updates, comprising:

means for classifying a set of requests;

means for queuing the set of requests having a fixed size;

means for interacting with a storage medium based upon at least one of a latency requirement and a throughput requirement.

29. A computer readable medium having a data structure stored thereon, comprising:

a first data field related to a size parameter for a queue associated with a round of requests; and

a second data field related to a latency parameter associated with the round of requests.